nodejs-express 환경 설정하기

전파고양이

November 7, 2021

Contents

1 config json으로 환경 변수 읽어오기

```
npm i config
```

```
curl localhost:4000
```

```
hello word
```

2 pm2-dev를 이용해서 hot-reloading 사용하기

package.json의 scripts에 아래와 같이 서버 Script를 넣어준다. NODE_ENV 환경 변수를 PHASE로 받아서 할당한다. timstamp option을 넣으면 log가 실행된다.

```
{
    "scripts": {
        "server": "NODE_APP_INSTANCE='' DANGEROUSLY_DISABLE_HOST_CHECK=true NODE_ENV=${PHASE} pm2-dev --timestamp start app.js",
        "test": "echo \"Error: no test specified\" && exit 1"
    }
}
```

3 typescript

```
tsc --init
```

위의 커맨드를 실행하면 tsconfig.json이 생성된다.

```
{
  "compilerOptions": {
    /* Visit https://aka.ms/tsconfig.json to read more about this file */

    /* Basic Options */
    // "incremental": true,                         /* Enable incremental compilation */
    "target": "es5",                                /* Specify ECMAScript target version: 'ES3' (default), 'ES5', 'ES2015', 'ES2016', 'ES2017',
```

```
    "module": "commonjs",                                /* Specify module code generation: 'none', 'commonjs', 'amd', 'system', 'umd', 'es2015
    // "lib": [],                                        /* Specify library files to be included in the compilation. */
    // "allowJs": true,                                  /* Allow javascript files to be compiled. */
    // "checkJs": true,                                  /* Report errors in .js files. */
    // "jsx": "preserve",                                /* Specify JSX code generation: 'preserve', 'react-native', 'react', 'react-jsx' or '
    // "declaration": true,                              /* Generates corresponding '.d.ts' file. */
    // "declarationMap": true,                           /* Generates a sourcemap for each corresponding '.d.ts' file. */
    // "sourceMap": true,                                /* Generates corresponding '.map' file. */
    // "outFile": "./",                                  /* Concatenate and emit output to single file. */
    // "outDir": "./",                                   /* Redirect output structure to the directory. */
    // "rootDir": "./",                                  /* Specify the root directory of input files. Use to control the output directory stru
    // "composite": true,                                /* Enable project compilation */
    // "tsBuildInfoFile": "./",                          /* Specify file to store incremental compilation information */
    // "removeComments": true,                           /* Do not emit comments to output. */
    // "noEmit": true,                                   /* Do not emit outputs. */
    // "importHelpers": true,                            /* Import emit helpers from 'tslib'. */
    // "downlevelIteration": true,                       /* Provide full support for iterables in 'for-of', spread, and destructuring when targ
    // "isolatedModules": true,                          /* Transpile each file as a separate module (similar to 'ts.transpileModule'). */

    /* Strict Type-Checking Options */
    "strict": true,                                      /* Enable all strict type-checking options. */
    // "noImplicitAny": true,                            /* Raise error on expressions and declarations with an implied 'any' type. */
    // "strictNullChecks": true,                         /* Enable strict null checks. */
    // "strictFunctionTypes": true,                      /* Enable strict checking of function types. */
    // "strictBindCallApply": true,                      /* Enable strict 'bind', 'call', and 'apply' methods on functions. */
    // "strictPropertyInitialization": true,             /* Enable strict checking of property initialization in classes. */
    // "noImplicitThis": true,                           /* Raise error on 'this' expressions with an implied 'any' type. */
    // "alwaysStrict": true,                             /* Parse in strict mode and emit "use strict" for each source file. */

    /* Additional Checks */
    // "noUnusedLocals": true,                           /* Report errors on unused locals. */
    // "noUnusedParameters": true,                       /* Report errors on unused parameters. */
    // "noImplicitReturns": true,                        /* Report error when not all code paths in function return a value. */
    // "noFallthroughCasesInSwitch": true,               /* Report errors for fallthrough cases in switch statement. */
    // "noUncheckedIndexedAccess": true,                 /* Include 'undefined' in index signature results */
    // "noImplicitOverride": true,                       /* Ensure overriding members in derived classes are marked with an 'override' modifie
    // "noPropertyAccessFromIndexSignature": true,       /* Require undeclared properties from index signatures to use element accesses. */

    /* Module Resolution Options */
    // "moduleResolution": "node",                       /* Specify module resolution strategy: 'node' (Node.js) or 'classic' (TypeScript pre-
    // "baseUrl": "./",                                  /* Base directory to resolve non-absolute module names. */
    // "paths": {},                                      /* A series of entries which re-map imports to lookup locations relative to the 'baseU
    // "rootDirs": [],                                   /* List of root folders whose combined content represents the structure of the projec
    // "typeRoots": [],                                  /* List of folders to include type definitions from. */
    // "types": [],                                      /* Type declaration files to be included in compilation. */
    // "allowSyntheticDefaultImports": true,             /* Allow default imports from modules with no default export. This does not affect cod
    "esModuleInterop": true,                             /* Enables emit interoperability between CommonJS and ES Modules via creation of names
    // "preserveSymlinks": true,                         /* Do not resolve the real path of symlinks. */
    // "allowUmdGlobalAccess": true,                     /* Allow accessing UMD globals from modules. */

    /* Source Map Options */
    // "sourceRoot": "",                                 /* Specify the location where debugger should locate TypeScript files instead of sourc
    // "mapRoot": "",                                    /* Specify the location where debugger should locate map files instead of generated lo
    // "inlineSourceMap": true,                          /* Emit a single file with source maps instead of having a separate file. */
    // "inlineSources": true,                            /* Emit the source alongside the sourcemaps within a single file; requires '--inlineSo

    /* Experimental Options */
    // "experimentalDecorators": true,                   /* Enables experimental support for ES7 decorators. */
    // "emitDecoratorMetadata": true,                    /* Enables experimental support for emitting type metadata for decorators. */
```

```
    /* Advanced Options */
    "skipLibCheck": true,                          /* Skip type checking of declaration files. */
    "forceConsistentCasingInFileNames": true       /* Disallow inconsistently-cased references to the same file. */
  }
}
```

설정을 보면 크게 아래의 옵션들이 있다.

- Basic Option
- Strict TypeChecking Options
- Additional Check
- Module Resolution Option
- Advanced Option

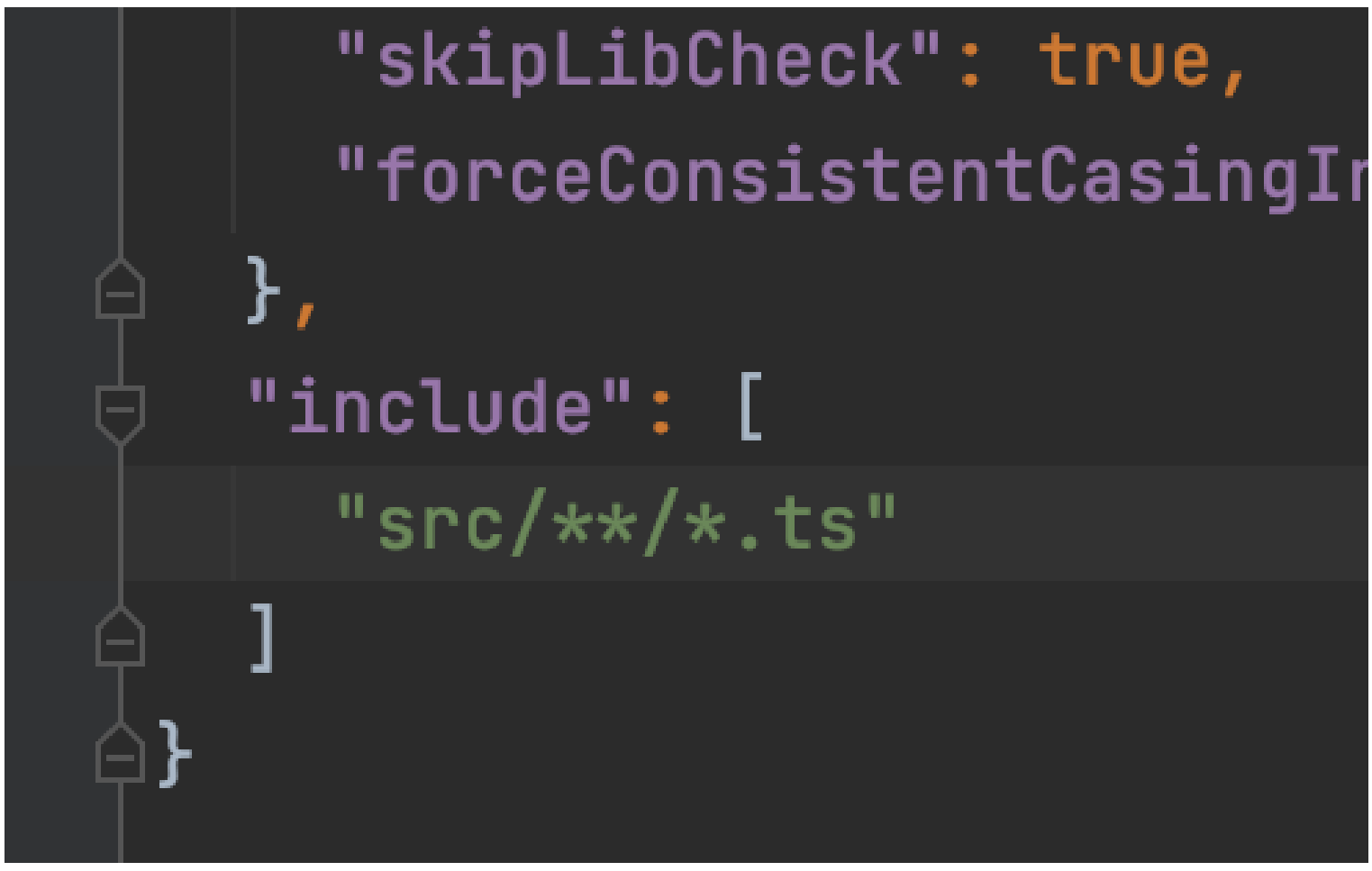
4 Tsconfig에 대해서 Document를 보려면 다음 링크에서 참고한다.

https://www.typescriptlang.org/tsconfig

5 pm2, node설치

npm install –save-dev pm2 ts-node

6 Tsconfig에서 SourceMap?



위와 같이 설정하고 나면 tsc만 실행해도 build경로에 파일들이 들어온다.

7 특정 파일을 빌드하고 싶을 때

tsc {fileName}

8 Trouble

8.1 type error

tsconfig에 아래와 같이 설정 되어 있었다.

```
"esModuleInterop": true
```

```
3 const app: express.Application = express();
                                   ~~~~~~~


  src/index.ts:1:1
    1 import * as express from 'express';
      ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

    Type originates at this import. A namespace-style import cannot be called or constructed, and will cause a failure at runtime. Conside
```

8.2 tsc-watch Typescript에서 Node-Express Hot Reloading

```
pm2-dev --interpreter ./node_modules/.bin/ts-node src/index.ts
```

```
2021-11-06-23:57:45 index-0  | internal/modules/cjs/loader.js:888
2021-11-06-23:57:45 index-0  |   throw err;
2021-11-06-23:57:45 index-0  |   ^
2021-11-06-23:57:45 index-0  | Error: Cannot find module 'typescript'
2021-11-06-23:57:45 index-0  | Require stack:
```

- solution
  - 왜 인지 모르겠는데 pm2 interpreter설정이 현재 경로의 node_moduls에 있는 파일들만 보는 것 같음.

```
npm install typescript --save-dev
```

8.3